A Decision Tree View of ReLU Neural Networks

1. Introduction

ReLU neural networks are typically described as **piecewise linear functions**. While correct, this description hides an important structural property: the network performs **conditional selection over a combinatorially large set of linear transformations**.

A more revealing interpretation is to view a ReLU network as an **implicit decision tree**, where each path corresponds to a specific linear operator constructed from shared parameters. This perspective clarifies both the expressive power and the internal organization of such networks.

---

2. ReLU Layers as Conditional Linear Operators

Consider a standard feedforward ReLU network. Each layer can be written as:

$x_{k+1} = D_k W_k x_k$

where:

- W_k is the weight matrix,
- D_k is a diagonal matrix of 0/1 entries induced by ReLU activations.

The matrix D_k acts as a **decision matrix**, selecting which components of the linear transformation (row selection) are active.

---

3. Synthetic Matrices

Including the weights in the next layer you have:

$W_{k+1} D_k W_k x_k$

Applying D_k to W_{k+1} instead of W_k results in a conditional synthetic matrix S.

$S = W_{k+1} D_k$

S then is a (locality sensitive) hash table lookup of a linear mapping $W_{effective}$ via column selection from the binary key in D.

A ReLU neural network then is:

$y=S_n..S_2S_1Wx$

Because each D_k is binary-valued, the number of possible synthetic matrices grows exponentially with layer width. However, all of them are constructed from the same shared weights.

---

4. Decision Tree Interpretation

This structure naturally induces a decision tree:

- Each layer corresponds to a **tree depth**.
- Each activation pattern (D) at that layer corresponds to a **branching decision**.
- Each branch selects a particular **linear operator fragment**.

At depth k, the network has implicitly defined a large set of possible intermediate linear mappings. Moving deeper into the network corresponds to:

> selecting a child node (a conditional linear mapping) and applying it to the current state.

Thus, a forward pass can be viewed as:

1. Start at the root with input x,
2. At each layer:
    - evaluate ReLU conditions,
    - select a child node (particular instance of S using key D),
    - apply it,
3. Continue until a leaf is reached.

Each **root-to-leaf path** defines one sequence of synthetic matrices.

---

5. Key Distinction from Classical Decision Trees

Unlike classical decision trees:

- Child nodes do not store constants but **linear transformations**.

- The number of child nodes is **exponentially large but implicit**.
- All child nodes share parameters inside the weight matrices.

This makes the structure far more compact and expressive than an explicit tree.

---

6. Implications

This decision tree view highlights several important properties:

- **Exponential expressivity**: A network with modest size can represent an enormous number of distinct linear mappings.
- **Parameter sharing**: All synthetic matrices reuse the same weights, enabling generalization.
- **Hierarchical conditioning**: Each layer refines the selection of the final transformation.

---

7. Conclusion

A ReLU neural network is best understood not just as a piecewise linear function, but as:

> an implicit, exponentially large decision tree whose leaves are linear operators constructed through shared compositions.

Each input traverses a unique path through this tree, successively selecting and applying conditional linear mappings, ultimately synthesizing a transformation tailored to that input.